**Subject: Zoo garden registers animals and their diets. Animals have specific diets (sometimes individual) and their caregivers should register any modifications**

**Customer**: Local ZOO

**Users**: Employees, who:

* register new animals
* create diets for animals
* give meal to animals

**Purpose**: Control diet of animals, registered in the ZOO.

**Project details**

CEO of the ZOO wants animals live their full lifecycle, and visitors could bring children to familiarize them with the local and exotic animals. To reach that purpose, everything in zoo has to be well organized, and each animal should be healthy.

Current database will be used in order to organize and control animal’s registration, location, special diets and changes that could be made during each feeding.

**Scenarios**

* New animal has been born and now has to be registered. Employee should check if there is enough space in the cage, where he want to settle it down.
* Animal is ill and his diet needs changes on this period, responsible employee creates new diet and assign it to the animal.
* Employee do not have current ingredient and has to give a substitution and register this action.
* Animal is pregnant and needs additional meal.
* Animal has died and it has to be registered.

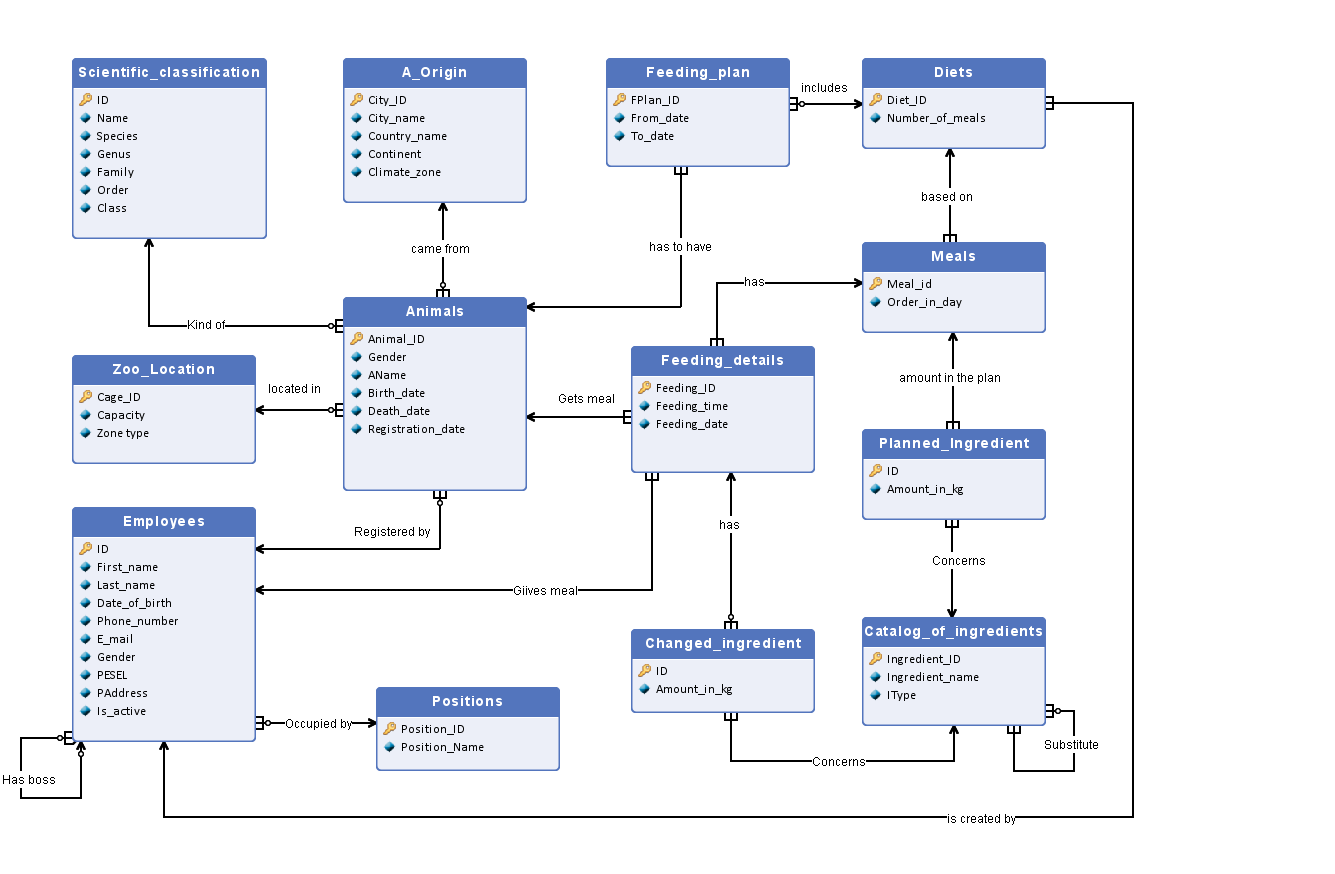
**Assumptions and limitations**

* Each animal is in his cage and there is no situation of overcrowding. All cages are marked.
* There is no situation that animal does not get food at all.
* The database does not contain info about the aspect of hygiene of animals.
* Does not contain information about salaries of employees.
* The database does not contain information about the health indicators of the animal, the history of treatment and the conclusions of the veterinarian.
* The database does not contain information about tickets and their prices that are selling to the visitors.

**Inquiries to the database:**

* Employee wants to check how many new animals has been registered in the ZOO during last month.
* Employee wants to check if ZOO has an animal of selected species or kingdom.
* Veterinarian wants to see which diet changes were made for the particular animal during last week.
* Employee wants to check how many diets contain grain.
* Employee wants to see from which country animal has arrived.
* Employee wants to check how many empty cages in the ZOO, their ID’s and location (zone type in this case).
* Employee wants to see all substitutes for cabbage.
* Employee wants to see the most popular diet (In how many feeding plans diet is included)
* Veterinarian wants to see how many diets have pregnant elephant and giraffes.
* Employee wants to see how many animals currently live in tropical zone\_type.
* Veterinarian wants to check how many calories lions have in their feeding plan.
* CEO wants to calculate how many kilograms of vegetables are needed for the nearest month

**ERD diagram**

****

**Description of Entities and attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: Animals**  Contains “personal” information about each animal, located in the ZOO. New entry is added each time new animal arrives to the ZOO. After 5 years after death, entry can be deleted. | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| Animal\_ID | integer | yes | Unique animal’s number in the ZOO. It is an integer from 1 to 29999 |
| Gender | VARCHAR | no | Represents gender, can have value “M” or “F”. |
| AName | VARCHAR | no | Animal’s name. Text string from 2 to 20 chars. Roman alphabet with polish signs. Any special characters not allowed (e.g. @,!,&) |
| Birth\_date | VARCHAR | no | Animal’s birth date. Can be NULL if unknown. Date type: dd.mm.yyyy |
| Death\_date | VARCHAR | no | Animal’s death date. Is NULL if it is alive. Date type: dd.mm.yyyy |
| Registration\_date | date | no | Day of appearing in the ZOO. Cannot be earlier than birth date. Date type: dd.mm.yyyy |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: A\_Origin**  Shows where animal came from. New entry is added when new animal arrived from the city, which does not exist in the database. Max number: 50000 | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| City\_ID | integer | yes | Unique city number. It is an integer from 1 to 50000. |
| City\_name | VARCHAR | no | Text string from 2 to 85 chars.  Roman alphabet with polish signs.  Any special characters not allowed (e.g. @,!,&) |
| Country\_name | VARCHAR | no | Text string from 2 to 56 chars.  Roman alphabet with polish signs.  Any special characters not allowed (e.g. @,!,&) |
| Continent | VARCHAR | no | Text string from 5 to 13 chars.  Roman alphabet with polish signs.  Any special characters not allowed (e.g. @,!,&) |
| Climate\_zone | VARCHAR | no | Text string from 3 to 11 chars.  Roman alphabet with polish signs.  Any special characters not allowed (e.g. @,!,&) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: Scientific\_classification**  Gives exhaustive details about animal’s nature. New entry is added when registers a new animal of such species that does not exist in the database. Max number: 29999 | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| ID | integer | yes | Unique number. It is an integer from 1 to 29999. |
| Name | VARCHAR | no |  |
| Species | VARCHAR | no | Name of the Species. Text string from 3 to 199 chars. Roman and Latin alphabet. Any special characters not allowed (e.g. @,!,&) |
| Genus | VARCHAR | no | Name of genus. Text string from 3 to 199 chars. Roman and Latin alphabet. Any special characters not allowed (e.g. @,!,&) |
| Family | VARCHAR | no | Name of family. Text string from 3 to 199 chars. Roman and Latin alphabet. Any special characters not allowed (e.g. @,!,&) |
| Order | VARCHAR | no | Name of the order (Class divides into smaller orders). Text string from 3 to 199 chars. Roman and Latin alphabet. Any special characters not allowed (e.g. @,!,&) |
| Class | VARCHAR | no | Name of one of seven classes. Text string from 3 to 199 chars. Roman and Latin alphabet. Any special characters not allowed (e.g. @,!,&) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: Zoo\_location**  Gives information about location and zone type, where animal is. Also shows how many animals in this location are. Max number: 4000 | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| Cage\_ID | integer | yes | Unique number of the cage. It is an integer from 1 to 4000. |
| Capacity | integer | no | Shows how many animals can be in this cage. It is an integer from 1 to 100 |
| Zone\_type | VARCHAR | no | Shows which conditions are created in this cage. (Temperature, humidity etc.)  Text string from 3 to 11 chars.  Roman alphabet with polish signs.  Any special characters not allowed (e.g. @,!,&) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: Employees**  Gives information about employees, working in the ZOO. New entry is added, when new person is employed. Cannot be deleted during 30 years after firing. Max number is 100000 | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| ID | integer | yes | Unique number of employee. It is an integer from 1 to 100000. |
| First\_name | VARCHAR | no | Name of the employee. Text string up to 30 chars. Roman alphabet with polish signs, any special characters not allowed (e.g. @,!,&) |
| Last\_name | VARCHAR | no | Surname of the employee. Text string up to 40 chars. Roman alphabet with polish signs, any special characters not allowed (e.g. @,!,&) |
| Date\_of\_birth | VARCHAR | no | Employee’s birth date. Date type: dd.mm.yyyy.  Allows to introduce date from the interval (1.01.1930-today) |
| Phone\_number | VARCHAR | no | Employee’s phone number. Contains 11 digits. Any special signs are not allowed. |
| E-mail | VARCHAR | no | String up to 30 chars, roman alphabet with special signs. Can be NULL |
| Gender | VARCHAR | no | Represents gender, can have value “M” or “F”. |
| PESEL | VARCHAR | no | A Polish identity number, consisting of 11 digits, out of which first 6 represent birth date. |
| PAddress | VARCHAR | no | Employee’s address. Text string up to 400 chars. |
| Is\_active | binary | no | 0 if not working, 1 if currently works in ZOO. Possible values: 0 or 1. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: Positions**  Gives information about employee’s position and subordination. Cannot be deleted during 30 years after firing(last date\_of\_end for the employee in this entity). Max number is 1000000 | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| Position\_ID | integer | yes | Unique number of position. It is an integer from 1 to 1000000. |
| Position\_name | VARCHAR | no | Name of the occupied position. Text string up to 50 chars. Roman alphabet with polish signs, any special characters not allowed (e.g. @,!,&) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: Catalog\_of\_ingredients**  Lists all ingredients that can be used in a diet. Up to 10000 lines | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| Ingredient\_ID | integer | yes | Unique number of ingredient. It is an integer from 1 to 10000. |
| Ingredient\_name | VARCHAR | no | Name of the ingredient. Text string up to 50 chars. Roman alphabet with polish signs, any special characters not allowed (e.g. @,!,&) |
| IType | VARCHAR | no | Type of the ingredient (e.g. vegetable, fruit, grain, etc.). Text string up to 30 chars. Roman alphabet with polish signs, any special characters not allowed (e.g. @,!,&) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: Planned\_Ingredient**  Describes planed amount of a particular ingredient in the meal | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| ID | integer | yes | Unique ID. It is an integer from 1 to 5000000 |
| Amount\_in\_kg | float | no | Amount of chosen ingredient in kilograms. float from 1.0 to 10000.0 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: Meals**  Describes what is planned to give to an animal in a particular meal. New entry is added, when new meal is created and there was no such before. Can have about 1000000 entries | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| Meal\_id | integer | yes | Unique number of meal. It is an integer from 1 to 1000000. |
| Order\_in\_day | integer | no | Which meal is it in the particular day |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: Diets**  Describes which diet is planned for the animal. New entry is added, when new diet is created. Can have about 2000000 entries | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| Diet\_ID | integer | yes | Unique number of diet. It is an integer from 1 to 2000000. |
| Number\_of\_meals | integer | no | Number of planned meals per day. Integer from 1 to 8 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: Feeding\_plan**  Describes time period for the particular diet to the particular animal. New entry is added every time animal gets new diet. An entry can be deleted after two years after its end date. Can have up to 5000000 entries | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| FPlan\_ID | integer | yes | Unique number feeding plan. It is an integer from 1 to 5000000. |
| From\_date | VARCHAR | no | Date type: dd.mm.yyyy |
| To\_date | VARCHAR | no | Date type: dd.mm.yyyy, cannot be earlier than from\_date |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: Feeding\_Details**  Shows details of actual feeding. An entry can be deleted after two years after its end date. Can have up to 5000000 entries | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| Feeding\_ID | integer | yes | Unique number of feeding. It is an integer from 1 to 5000000. |
| Feeding\_Time | VARCHAR | no | Time of feeding. Format digits with one special sign allowed “:” hh:mm (hours:minutes). |
| Feeding\_Date | VARCHAR | no | Date type: dd.mm.yyyy. Day of feeding |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity: Changed\_Ingredient**  Employee registers changes in particular meal using this entity. New entry is added, when employee changes planned ingredient to a substitution. An entry can be deleted after two years after its end date. Can have up to 5000000 entries | | | |
| **Attributes** | | | |
| **Name** | **Type** | **PK** | **Description** |
| ID | integer | yes | Unique ID of changes. It is an integer from 1 to 5000000. |
| Amount\_in\_kg | float | no | Amount of chosen ingredient in grams. Integer from 1 to 10000 |

**Definition of relationships between entities**

* Relationship ***Kind of* 1..0-n** between **Animal** and **Scientific classification** represents animal’s belonging to the species and classes. One animal is of definite species and many animals in the ZOO can be of the same species. It might be a situation, that currently in the ZOO there is no animal of a particular species that is why optionality added.
* Relationship ***Came from*** **1..0-n** between **Animal** and **Origin** represents where animal was born. One animal has exactly one hometown; there are zero or many animals in the ZOO, that have been born in the same city.
* Relationship ***Located in*** **1..0-n** between **Animal** and **Zoo location** shows cage number where animal is located. One cage can have zero or many animals, living animal can be in only one particular cage.
* Relationship ***Registered by*** **1..0-n** between **Animal** and **Employee**– represents who registered animal in the ZOO. One animal is registered by one employee, but employee can register zero (as not everybody do registration) or many animals during his work in the ZOO.
* Relationship ***Occupied by*** **1..0-n** between **Position** and **Employee** entities shows current position of an employee. Working employee has exactly one position and different employees can occupy same position.
* Relationship ***Is created by*** **1..1-n** between **Employee** and **Diet** shows who exactly created a diet. Diet has one creator, but employee can create many different diets.
* Relationship ***Gives meal* 1..1-n** between **Employee** and **Feeding\_Details** shows who fed current animal and when. Feeding details can be done by one employee, but one employee can do several feeding details.
* Relationship ***Gets meal* 1..1-n** between **Animal** and **Feeding\_Details** shows execution of the feeding plan. One Feeding detail concerns one particular animal during its feeding, but animal can have many feeding details per day.
* Relationship ***Has to have* 1..1-n** between **Animal** and **Feeding\_Plan** represents what plan of feeding animal has. One plan relates to one animal, animal has one or many feeding plans at ones. There is no option for animal not to have a feeding plan.
* Relationship ***Includes* 1..0-n** between **Diet** and **Feeding\_Plan** shows when particular diet is planed for the animal. Feeding plan created for one diet but same diet can be in zero or many different feeding plans (e.g. diet was created, and used in the past but there is no animal or circumstances for giving it now).
* Relationship ***Based on* 1..1-n** between **Diet** and **Meal** represents what meals are included in a diet. Meal is assigned to a particular diet, but diet consists of several meals. Diet cannot be without any meal.
* Relationship ***Amount in the plan* 1..1-n** between **Meal** and **Planed\_ Ingredient** shows what ingredients and of which amount are included in this meal. Meal contains of one or several different ingredients. Meal cannot be without any ingredients.
* Relationship ***Concerns* 1..1-n** between **Catalog\_of\_ingredients** and **Planed\_Ingredient** shows how much of ingredients will be used. Same ingredient can be taken many times in different amount.
* Relationship ***Has* 1..0-n** between **Feeding\_Details** and **Changed\_ingredient** shows all changes during feeding. Changes are registered in particular feeding details, feeding details may have zero (because not every feeding has changes) or many changes.
* Relationship ***Concerns* 1..1-n** between **Catalog\_of\_ingredients** and **Changed\_ingredient** represents which ingredient was taken and of what amount. Change applies to the one ingredient but one ingredient can be in different registered changes.
* Relationship ***Substitute* n..0-n** between **Catalog\_of\_ingredients** and **Catalog\_of\_ingredients** shows how different ingredients can be substituted. One ingredient can be a substitute for several others, some ingredients may have no substitutes.
* Relationship **Has 1..1-n** between Feeding details and Meals. Shows which meal is given in reality.

**Relational database schema**

* **Catalog\_of\_ingredients**(Ingridient\_ID, Ingredient\_name, IType)

(Ingridient\_ID) PK

**Substitute**(Ing1 REF Catalog\_of\_ingredients, Ing2 REF Catalog\_of\_ingredients)

(Ing1, Ing2) PK

(Ing1, Ing2) FK REF Catalog\_of\_ingredients

* **Planned\_Ingredient**(ID, Amount, M\_ID REF Meal, Ing\_ID REF Catalog\_of\_ingredients)

(ID, M\_ID, Ing\_ID) PK

(M\_ID) FK REFERENCED Meal

(Ing\_ID) FK REFERENCED Catalog\_of\_ingredients

* **Meal**(Meal\_ID, Calories, Count, DietID REF Diet)

(Meal\_ID) PK

(DietID) FK REFERENCED Diet

* **Diet**(Diet\_ID, Number\_of\_meals, Dietition\_ID REF Employee)

(Diet\_ID) PK

(Dietition\_ID) FK REFERENCED Employee

* **Origin**(City\_ID, City\_Name, Country\_name, Continent, Climate\_zone)

(City\_ID) PK

* **Scientific classification**(ID, Species, Genus, Family, Order, Class)

(ID) PK

* **Zoo location**(Cage\_ID, Capacity, Zone\_type)

(Cage\_ID) PK

* **Employees**(ID, First\_name, Last\_name, Date\_of\_birth, Phone\_number, E-mail, Gender, PESEL, PAddress, Is\_active, Manager\_ID REF Employees, Position\_ID REF Positions)

(ID) PK

(Manager\_ID) FK REFERENCES Employees

(Position\_ID) FK REFERENCES Position

* **Positions**(Position\_ID, Position\_Name)

(Position\_ID) PK

* **Animal**(Animal\_ID, Gender, Birth\_date, Name, Death\_date, Registration\_date, ID\_SC REF Scientific classification, CityID REF Origin, C\_ID REF Zoo location, ID\_E REF Employee )

(Animal\_ID) PK

(ID\_SC) FK REFERENCES Scientific classification

(CityID) FK REFERENCES Origin

(C\_ID) FK REFERENCES Zoo location

(ID\_E) FK REFERENCES Employee

* **Changed\_Ingredients**(ID, Amount\_in\_kg, Ing\_ID REF Catalog\_of\_ingredients, Feed\_ID REF Feeding\_Details)

(ID) PK

(Ing\_ID) REFERENCES Catalog\_of\_ingredients

(Feed\_ID) REFERENCES Feeding\_Details

* **Feedinf\_Plan**(FPlan\_ID, From\_date, To\_date, D\_ID REF Diet, A\_ID REF Animal)

(FPlan\_ID) PK

(D\_ID) FK REFERENCE Diet

(A\_ID) FK REFERENCE Animal

* **Feeding\_Details**(Feeding\_ID, Feeding\_Time, Feeding\_Date, E\_ID REF Employee, A\_ID REF Animal)

(Feeding\_ID) PK

(E\_ID) FK REFERENCE Employee

(A\_ID) FK REFERENCE Animal

(M\_ID) INT REFERENCE Meals